DOCUMENT RESUME

ED 097 019 IR 001 201

AUTHOR	Fitzhugh, Robert J.; Pethia, Richard D.
TITLE	Disk File Management in a Medium-Scale Time-Sharing System.
INSTITUTION	Pittsburgh Univ., Pa. Learning Research and Development Center.
SPONS AGENCY	National Inst. of Education (DHEW), Washington, D.C.; National Science Foundation, Washington, D.C.
REPORT NO	PU-LRDC-1974-4
PUB DATE	74
NOTE	14p.; Paper presented at the Digital Equipment Corporation Users Society Fall Symposium (San Francisco, California, November, 1973)
EDRS PRICE	MF-$0.75 HC-$1.50 PLUS POSTAGE
DESCRIPTORS	Computer Programs; *Computers; *Computer Science; *Computer Storage Devices; Management Systems; *Time Sharing
IDENTIFIERS	ETSS; *Experimental Time Sharing System; Learning Research and Development Center

ABSTRACT

The paper descibes a compact and highly efficient disk file management system responsible for the management and allocation of space on moving head disk drives in a medium-scale time-sharing system. The disk file management system is a major component of the Experimental Time-Sharing System (ETSS) developed at the Learning Research and Development Center. ETSS has been successfully operating for nearly two years and is a multilanguage general-purpose time-sharing system based on Digital Equipment Coporation's PDP-15. (Author)

BEST COPY AVAILABLE

LEARNING RESEARCH AND DEVELOPMENT CENTER

1974/4

DISK FILE MANAGEMENT IN A MEDIUM-SCALE TIME-SHARING SYSTEM

ROBERT J. FITZHUGH AND RICHARD D. PETHIA

ED 092019

IR 001 301

ED 092019

DISK FILE MANAGEMENT IN A MEDIUM-SCALE TIME-SHARING SYSTEM

Robert J. Fitzhugh and Richard D. Pethia

Learning Research and Development Center
University of Pittsburgh

Paper presented at the Digital Equipment Corporation Users Society Fall Symposium, San Francisco, California, November 27-30, 1973.

The research reported herein was supported by a grant from the National Science Foundation (NSF-GJ-540X) and by the Learning Research and Development Center, supported in part by funds from the National Institute of Education, United States Department of Health, Education, and Welfare. The opinions expressed do not necessarily reflect the position or policy of the sponsoring agencies and no official endorsement should be inferred.

U S DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

Abstract

This paper describes a compact and highly efficient disk file management system responsible for the management and allocation of space on moving head disk drives in a medium-scale time-sharing system. The disk file management system is a major component of the Experimental Time-Sharing System (ETSS) developed at the Learning Research and Development Center. ETSS has been successfully operating for nearly two years and is a multi-language, general-purpose time-sharing system based on a DEC PDP-15.

DISK FILE MANAGEMENT IN A MEDIUM-SCALE TIME-SHARING SYSTEM

Robert J. Fitzhugh and Richard D. Pethia

Learning Research and Development Center
University of Pittsburgh

Introduction

Most commercially available small-machine time-sharing systems offer rudimentary or special-purpose disk file management software. Some systems require the programmer to be aware of the underlying disk structure and often place the responsibility for record packing within allocated disk space upon the program. More commonly, the file management software is special-purpose and is tailored to meet the requirements of a single language such as BASIC or a special application such as hospital record keeping. The file management system described in this paper is a demonstration that a powerful, general-purpose disk file management system offering large system features and capabilities is possible on a smaller computer with only limited memory.

Called DFM for Directory File Management, the system is the successor to two earlier disk file management systems, each based on inverted list principles and designed for fixed-head disk environments. The current DFM employs list structures and is designed for moving head devices although fixed head units can be supported as well. DFM in its present form has been in use for ten months and is a major component of the Experimental Time-Sharing System (ETSS) developed at the University of Pittsburgh. ETSS has been operational for nearly two

years and is a multi-language, general-purpose time-sharing system based on a Digital Equipment Corporation PDP-15.

DFM maintains a complete and self-contained directory structure on each disk unit so that packs may be mounted and dismounted at will independent of all others. Packs are named providing the programmer with the option of specifying a pack by name or simply permitting the file operation to default to the one or more resident system packs. Consistent with the device independent features of ETSS, DFM allows users to read, write, and manipulate data stored in named datasets with no regard for the physical properties of the device. Each dataset may contain any number of variable length records separated by any number of file marks. Datasets may be temporary, or they may be "cataloged" with protection keys in the user's private directory, in the system "library" available to all or in the directory of another user if permitted. Features are provided to enable multiple users to simultaneously access and update common datasets without the risk of interference.

Description

Device Characteristics. In the current implementation, DFM operates on IBM 2314-equivalent disk drives using 2316 or equivalent disk packs. The hardware established minimum unit of transfer is a 256 18-bit word sector. There are 200 sectors per cylinder and 200 cylinders per pack for a total of 40,000 sectors or 10.24 million words. This translates to approximately 25 million characters of storage on a single pack. The average rotational delay of the unit is 12.5 milliseconds, and head seek times range from 20-80 milliseconds with an average of 50 milliseconds.

Although DFM was designed for 2314-equivalent units, the specific device details are important only on the device handler level. A variety of disks can be supported including fixed head as well as moving head units.

Space Allocation and Deallocation. It is clear from the device performance statistics that head movement is a major component of total transfer time and that excessive or unnecessary head movement will sharply degrade overall device performance. This is particularly true for the space allocation and deallocation processes. As a result, a primary design objective was to minimize the number of head movements required to locate available space and to return space that is no longer needed. In addition, the allocation algorithm was designed to allocate space in such a way as to minimize head movement during subsequent file processes.

The minimum unit of allocation and deallocation is an implementation-defined block which, in the current release of ETSS, is two disk sectors or 512 18-bit words. Two different types of bit maps are used to monitor the allocated/unallocated status of the 20,000 blocks on each pack. A "Cylinder Bit Map" is maintained on each cylinder which reflects the allocated/unallocated status of each block on that cylinder. As there are 100 blocks on a cylinder, the bit map requires 100 bits or only six 18-bit words. When a new pack is mounted, each of these cylinder bit maps is scanned and a memory resident "Bit Map of Bit Maps" is constructed for the pack. Each bit in this memory resident bit map reflects the status of a cylinder bit map and indicates whether or not there is unallocated space on the cylinder. Only 12 18-bit words are required for this bit map.

When a disk block is required, an attempt is made to allocate space from the cylinder on which the disk heads are currently positioned. If this is not possible and space must be found on another cylinder, the memory resident bit map of bit maps is examined to avoid unnecessary accesses of cylinder bit maps in which there is no unallocated space. The lowest numbered cylinder with unallocated space is selected, the heads are moved to that cylinder and the cylinder bit map is read into memory. The two levels of bit maps ensure that no more than one head movement is ever required to allocate space. The cylinder selection strategy eliminates the scattering of allocated blocks and reduces head movements during subsequent file processes.

Space deallocation is a background process and is permitted to occur only when no other disk file processes are active. When a block is to be deallocated, the appropriate cylinder bit map is read into memory, and the bit corresponding to the block is reset. The bit map of bit maps is updated also if the cylinder was fully allocated prior to the deallocation.

Primary Data Structure. The primary and sole data structure on disk is a noncircular bidirectional list of blocks with each block containing "records" linked in a circular, bidirectional list within the block. Shown in Figure 1, this data structure is used in both directories and in datasets and is uniform and consistent on all levels. As a result, DFM is relatively small in size because only a few, very compact subroutines are required to create and manage this single data structure.

The first two words of each block in a list serve as pointers to the preceding and following blocks. Bidirectionality enables block lists to be scanned in either direction and permits validity checks to be performed on all links during normal file operations. To eliminate the

BEST COPY AVAILABLE

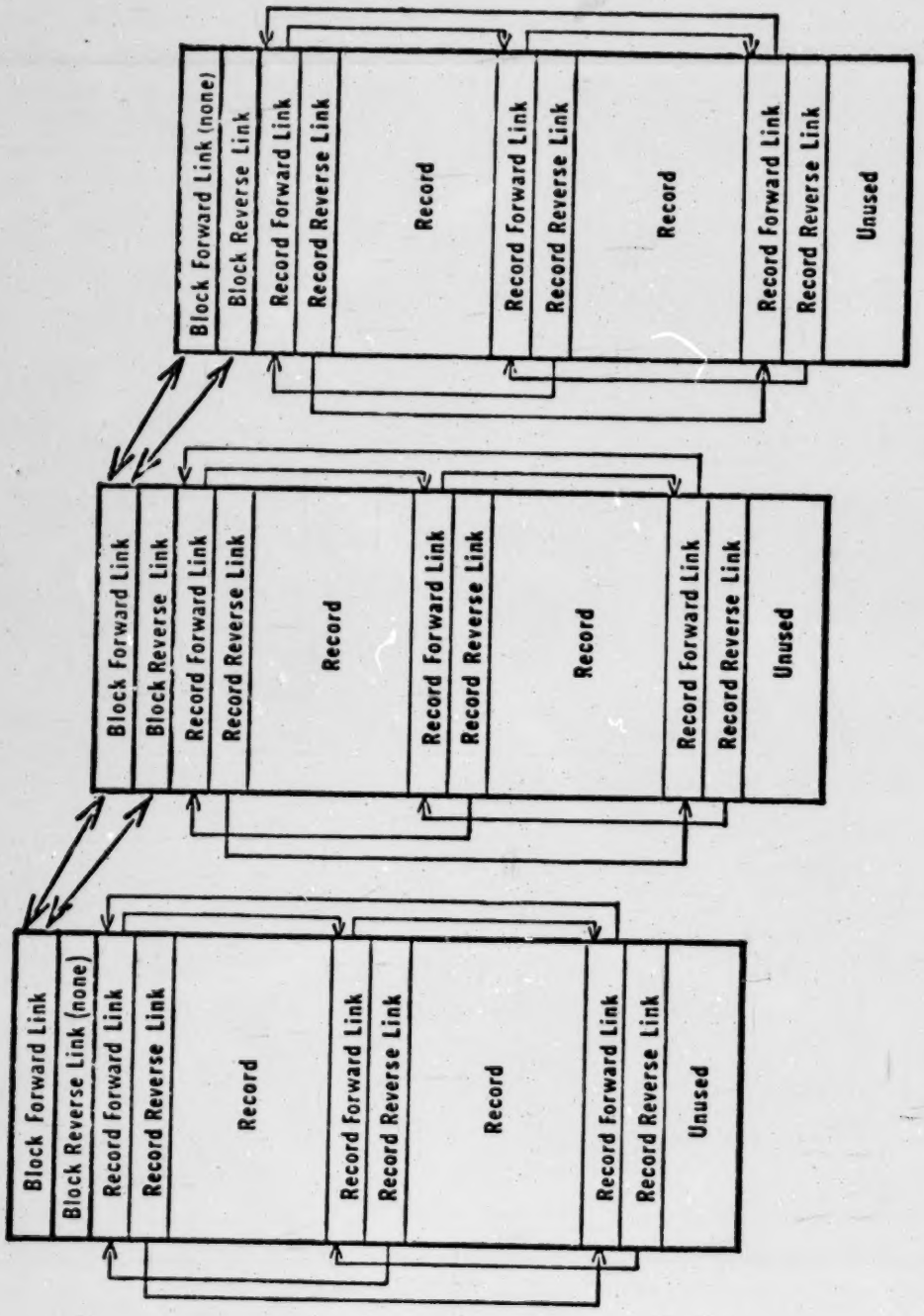

Figure 1. Primary data structure

requirement that the backward link of the first block in a list be updated each time a new block is added to the end of the list, block lists are noncircular. A null value is used to indicate that a block is either the first or the last block in a list.

Records within a block are linked in a circular, bidirectional list with each record preceded by a pair of list pointers. Circularity is possible since the entire block is read into memory and can be examined and updated at one time. Any unused space within a block appears as the last record in the list.

Directory Structure. The DFM directory structure consists of three interlinked levels and is depicted in Figure 2. For each pack, there is a single "Master Directory" of user ID's and other user specific information, "User Directories" of dataset names and other dataset specific information and "Datasets" containing "records" written to the disk by user programs. The master directory consists of one or more linked blocks that contain the identification numbers of all users who have user directories on the pack, other information about those users and pointers to each user directory. The ID of a user is added automatically to this master directory when the user catalogs a dataset on the pack. This ID entry is removed if all the datasets in the associated user directory are deleted.

A user directory is created when a new user catalogs a dataset on the pack and is automatically destroyed when all the datasets within the user directory have been deleted. As with the master directory, a user directory will expand and contract in size as datasets are cataloged or deleted. An entry in a user directory consists of a dataset name, information about the dataset and a pointer to the first block of the dataset. This entry is removed when the dataset is deleted by the user.

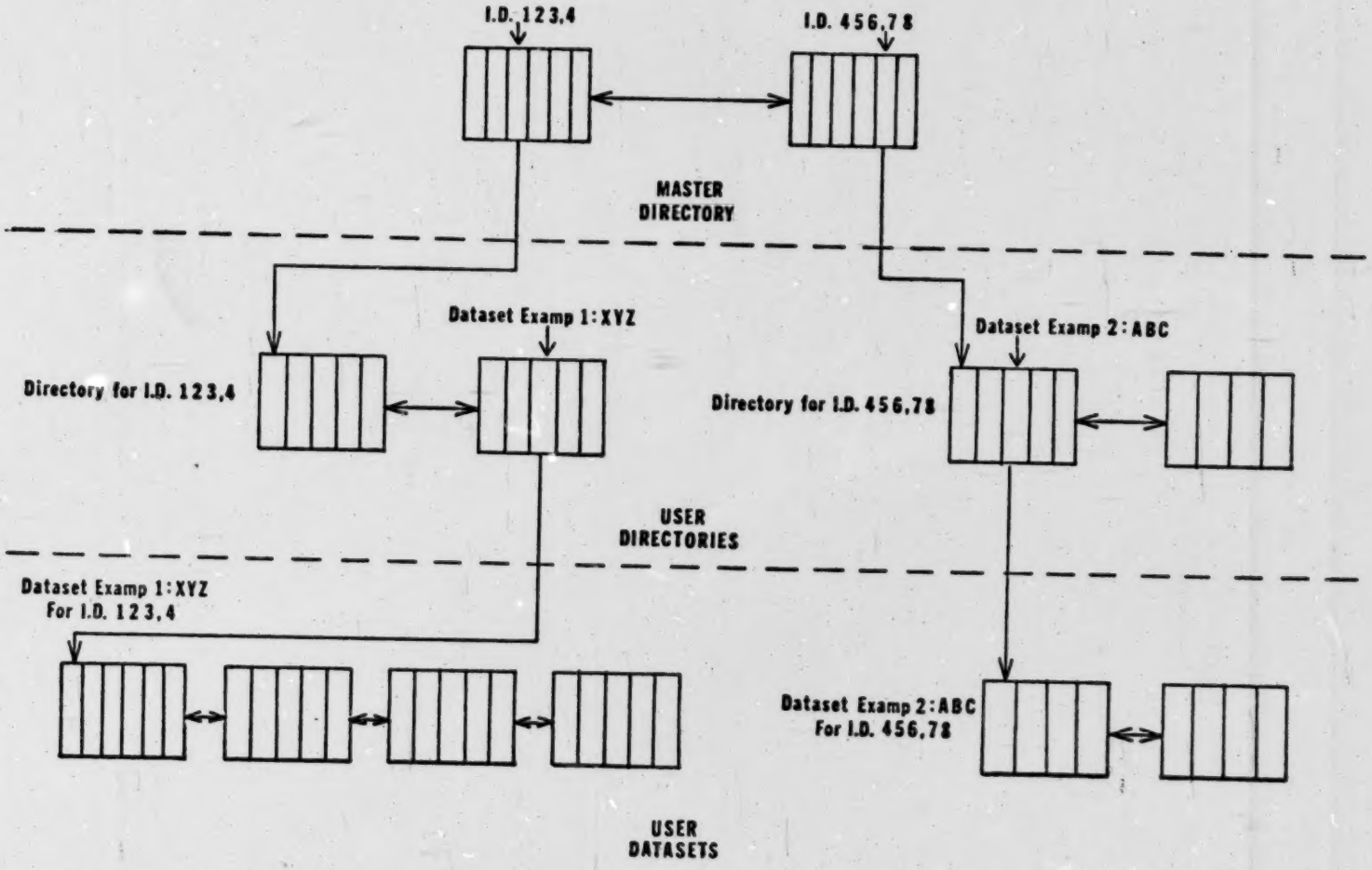

Figure 2. Directory structure

BEST COPY AVAILABLE

A dataset consists of a series of variable length records stored within blocks. Although a dataset may contain as many records as can be stored on a disk pack, no record can be longer than a single block less the link words. The manner in which records are stored within blocks is transparent to the user who merely reads and writes records completely unaware of the underlying block structure of the pack.

Operation

File Processing. When a program indicates that it wishes to access a cataloged dataset, the master directory of the specified pack is searched for the appropriate ID. If it is found, the pointer to the associated user directory is retrieved, and the user directory is searched for the dataset name. If the name appears in the directory, an entry is created and is added to a memory resident list called the "Active List." Each entry in the active list contains the name of a dataset currently being accessed, information about the dataset and the user accessing it, and a set of values identifying the current position of the user within the dataset. These include the block numbers of the prior, current and next blocks in the dataset and the position of the current record within the current block. These values are updated during normal file processing as the user moves within the dataset through file manipulation requests or by reading and writing records.

The memory resident active lists enables file requests to be satisfied quickly without time consuming directory or dataset searches. A read request, for example, requires only a single transfer. The memory resident entry uniquely identifies the disk block where the next record resides and the position of the record within the block. A write request usually requires only two transfers, the first to read in the current block and the second to rewrite it to disk with the newly added record.

Less frequently, when an additional block must be allocated, a write request will require from three to five transfers depending upon the cylinder bit map and disk block already in memory.

Dataset Deletion. When a dataset is to be deleted, the user directory entry for the dataset is removed first. If no names remain in the user directory, the complete user directory is deleted as well as the associated ID in the master directory. The list of blocks containing the dataset is then linked to the end of the disk resident delete list in preparation for the background deallocation process. From the user's point of view, the dataset is effectively deleted at this time, even though actual deallocation has not yet begun. During intervals in which no other requests are pending, successive blocks are delinked from the delete list, and the corresponding bits in the cylinder bit maps are reset.

Directory Integrity Following a Crash. The directory structure is relatively impervious to the effects of disk failures or system crashes. All directory creation and modification operations are sequenced so that an unexpected interruption will not result in directory inconsistencies or doubly allocated space. For example, when a new block must be allocated for a dataset, the new block is allocated and is written to disk with the appropriate backward link before the forward link of the prior block is updated. Were this sequence interrupted, the worst outcome would be allocated space that is outside the directory structure and is unaccounted for. The directory structure remains intact.

Although the directory structure could be disrupted by a hardware or power failure which interrupts an on-going disk transfer, operating experience with the current as well as prior versions of DFM has demonstrated that a well-designed disk structure can be made substantially crash-proof if directory creation and update operations are

properly sequenced. In nearly two years of daily operation, no dataset or directory has ever been disrupted or lost on ETSS.

Conclusion

DFM is an operating demonstration that a powerful, general-purpose disk file management system is feasible on a smaller computer with limited memory. In ten months of extensive use, the file management system has met or exceeded all major performance objectives and is currently managing over 7000 on-line datasets on the resident system packs. The primary data structure of lists of blocks and lists of records within blocks have proved to be sound and particularly well-suited for small-machine operating systems.

References

Fitzhugh, R. LRDC experimental time-sharing system reference manual. Pittsburgh: University of Pittsburgh, Learning Research and Development Center, 1970. 3 vols.

Fitzhugh, R. A general-purpose time-sharing system for a small- or medium-scale computer. Pittsburgh: University of Pittsburgh, Learning Research and Development Center, 1973. Publication 1973/23.

Fitzhugh, R. Laboratory control with a medium scale time-sharing system. Pittsburgh: University of Pittsburgh, Learning Research and Development Center, 1973. Publication 1973/25.